ASSIGNMENT- 4.3

2303A51234

AkHILA CHETI

BATCH – 18

Task 1: Zero-Shot Prompting – Leap Year Check

Scenario

Zero-shot prompting involves giving instructions without providing examples.

Task Description

Use zero-shot prompting to instruct an AI tool to generate a Python function that:

- Accepts a year as input
- Checks whether the given year is a leap year
- Returns an appropriate result

Note: No input-output examples should be provided in the prompt.

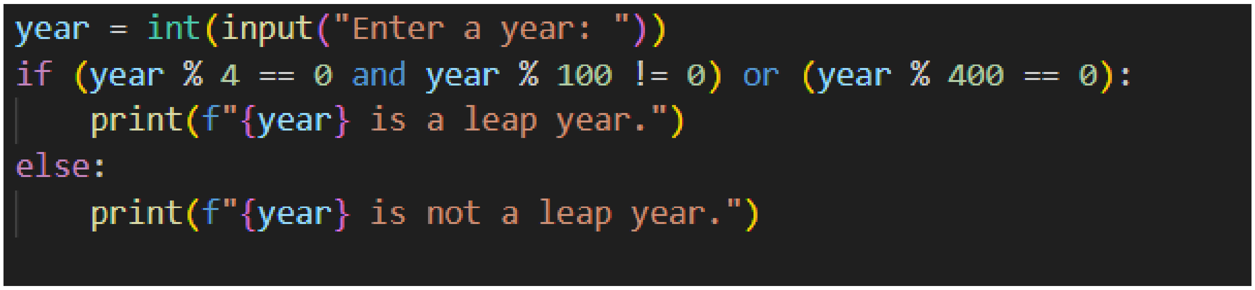
Expected Output

- AI-generated leap year checking function
- Correct logical conditions
- Sample input and output
- Screenshot of AI-generated response (if required)

#PROMPT:

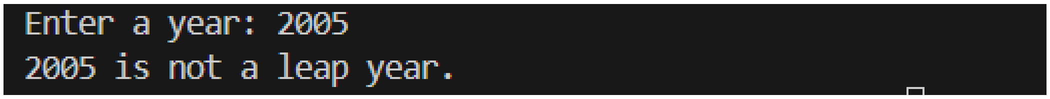
#check whether the year is a leap year or not taking input from the user

CODE:



```
year = int(input("Enter a year: "))
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
    print(f"{year} is a leap year.")
else:
    print(f"{year} is not a leap year.")
```

OUTPUT:



```
Enter a year: 2005
2005 is not a leap year.
```

Analysis:

1.AI generated correct code.

2.Output was accurate.

3. Easy to use in vs.

Task 2: One-Shot Prompting – Centimeters to Inches Conversion

Scenario

One-shot prompting guides AI using a single example.

Task Description

Use one-shot prompting by providing one input-output example to generate a Python

function that:

- Converts centimeters to inches
- Uses the correct mathematical formula

Example provided in prompt:

Input: 10 cm → Output: 3.94 inches

Expected Output

- Python function with correct conversion logic
- Accurate calculation
- Sample test cases and outputs

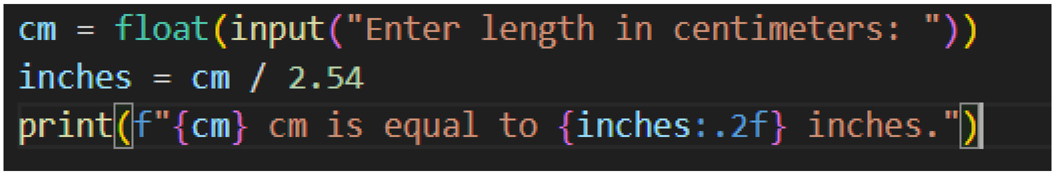
#prompt:

#convert centimeters to inches output should be in 2 decimal places

#Input: 10 cm

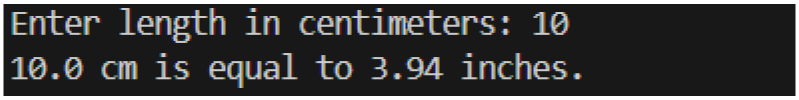
# Output: 3.94 inches

Code:



```python
cm = float(input("Enter length in centimeters: "))
inches = cm / 2.54
print(f"{cm} cm is equal to {inches:.2f} inches.")
```

### Output:



```
Enter length in centimeters: 10
10.0 cm is equal to 3.94 inches.
```

Analysis:

- One example helped the AI generate correct code.
- The task improved understanding of AI pattern learning.
- Clear prompts gave accurate output.

Task 3: Few-Shot Prompting – Name Formatting

Scenario

Few-shot prompting improves accuracy by providing multiple examples.

Task Description

Use few-shot prompting with 2–3 examples to generate a Python function that:

• Accepts a full name as input

• Formats it as “Last, First”

Example formats:

- "John Smith" → "Smith, John"
- "Anita Rao" → "Rao, Anita"

Expected Output

- Well-structured Python function
- Output strictly following example patterns
- Correct handling of names
- Sample inputs and outputs

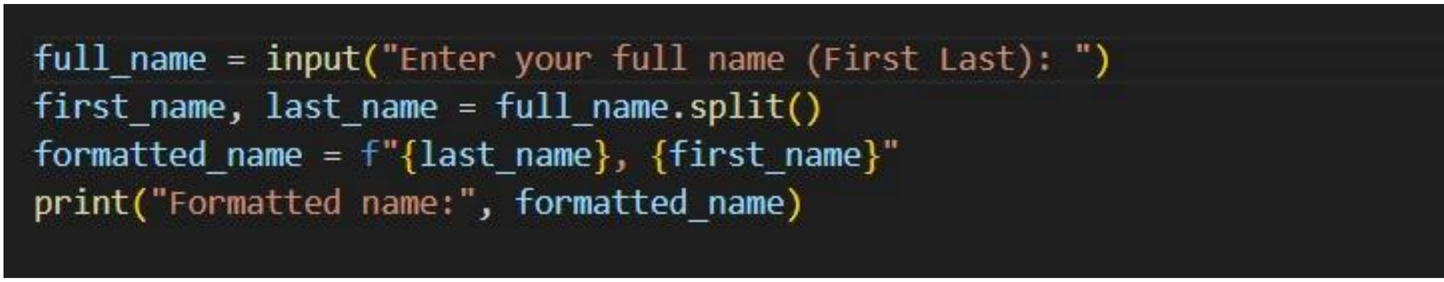
#prompt:

#name formatting input first name and last name and output should be in the format Last name, First name

#"John Smith" → "Smith, John"

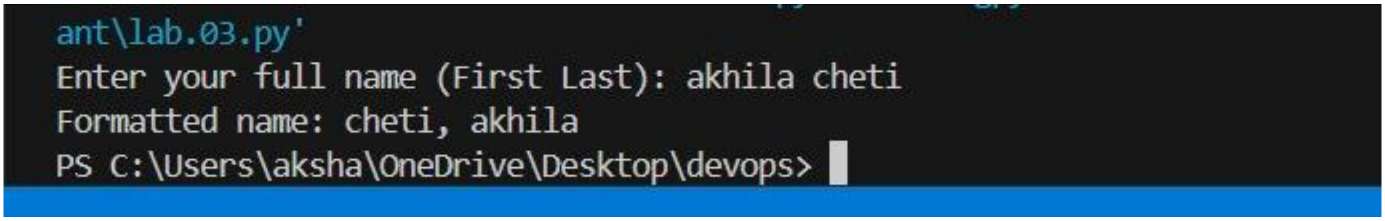
#"Anita Rao" → "Rao, Anita"

Code:



```
full_name = input("Enter your full name (First Last): ")
first_name, last_name = full_name.split()
formatted_name = f"{last_name}, {first_name}"
print("Formatted name:", formatted_name)
```

Output:



```
ant\lab.03.py'
Enter your full name (First Last): akhila cheti
Formatted name: cheti, akhila
PS C:\Users\aksha\OneDrive\Desktop\devops>
```

Analysis:

- examples helped the AI understand name formatting.
- The task improved accuracy in code generation.
- Clear prompts produced correct output.

Task 4: Comparative Analysis – Zero-Shot vs Few-Shot

Scenario

Different prompt strategies may produce different code quality.

Task Description

- Use zero-shot prompting to generate a function that counts vowels in a string
- Use few-shot prompting for the same problem
- Compare both outputs based on:

o Accuracy

o Readability

o Logical clarity

Expected Output

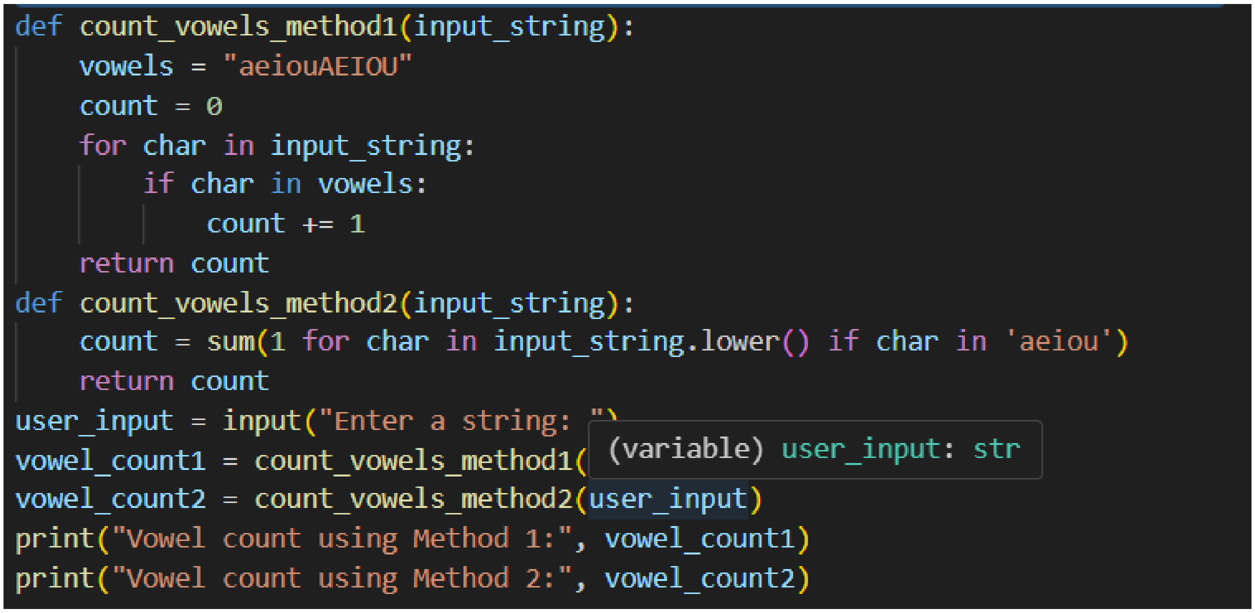
- Two vowel-counting functions
- Comparison table or short reflection paragraph
- Conclusion on prompt effectiveness

#prompt:

# code to count vowels in a string.

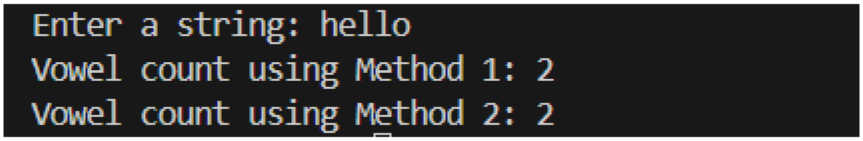
# Compare both functions based on:Accuracy, Readability, and Logical Clarity

Code:



```
def count_vowels_method1(input_string):
    vowels = "aeiouAEIOU"
    count = 0
    for char in input_string:
        if char in vowels:
            count += 1
    return count
def count_vowels_method2(input_string):
    count = sum(1 for char in input_string.lower() if char in 'aeiou')
    return count
user_input = input("Enter a string: ")
vowel_count1 = count_vowels_method1(
vowel_count2 = count_vowels_method2(user_input)
print("Vowel count using Method 1:", vowel_count1)
print("Vowel count using Method 2:", vowel_count2)
```

Output:



```
Enter a string: hello
Vowel count using Method 1: 2
Vowel count using Method 2: 2
```

Analysis:

- ai made task easy and accurate
- complicated codes can be easily executed
- various methods can be performed accurately

Task 5: Few-Shot Prompting – File Handling

Scenario

File processing requires clear logical understanding.

Task Description

Use few-shot prompting to generate a Python function that:

- Reads a .txt file
- Counts the number of lines in the file
- Returns the line count

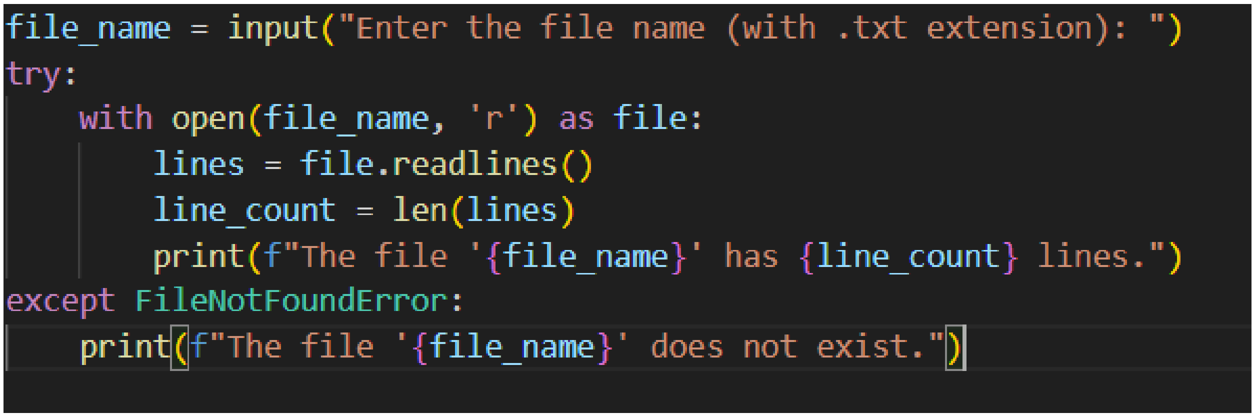
Expected Output

- Working Python file-processing function
- Correct line count
- Sample .txt input and output
- AI-assisted logic explanation

#prompt:

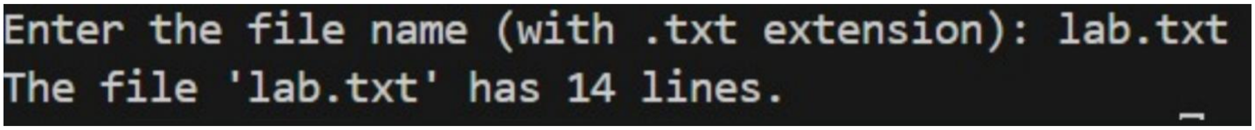
#read a .txt file and count number of lines in file and return file count

Code:



```python
file_name = input("Enter the file name (with .txt extension): ")
try:
    with open(file_name, 'r') as file:
        lines = file.readlines()
        line_count = len(lines)
        print(f"The file '{file_name}' has {line_count} lines.")
except FileNotFoundError:
    print(f"The file '{file_name}' does not exist.")
```

Output:



```
Enter the file name (with .txt extension): lab.txt
The file 'lab.txt' has 14 lines.
```

Analysis:

- vs code made reading file easy
- the complex codes are executed accurately
- clear prompts produced with correct outputs